

(19) 日本国特許庁 (J P)

## (12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2003-36164

(P 2 0 0 3 - 3 6 1 6 4 A)

(43) 公開日 平成15年2月7日 (2003. 2. 7)

(51) Int. Cl. <sup>7</sup>	識別記号	F I	テ-マコード (参考)
G06F 3/14	350	G06F 3/14	A 5B069
G09G 5/14		G09G 5/14	C 5C082

審査請求 未請求 請求項の数 3 O L (全7頁)

(21) 出願番号 特願2001-222874 (P 2001-222874)

(22) 出願日 平成13年7月24日 (2001. 7. 24)

(71) 出願人 000004329

日本ビクター株式会社

神奈川県横浜市神奈川区守屋町3丁目12番地

(72) 発明者 大竹 信二

神奈川県横浜市神奈川区守屋町3丁目12番地 日本ビクター株式会社内

(74) 代理人 100083806

弁理士 三好 秀和 (外9名)

Fターム (参考) 5B069 AA01 BA03 BB16 CA02 CA03

CA06 CA14 CA17

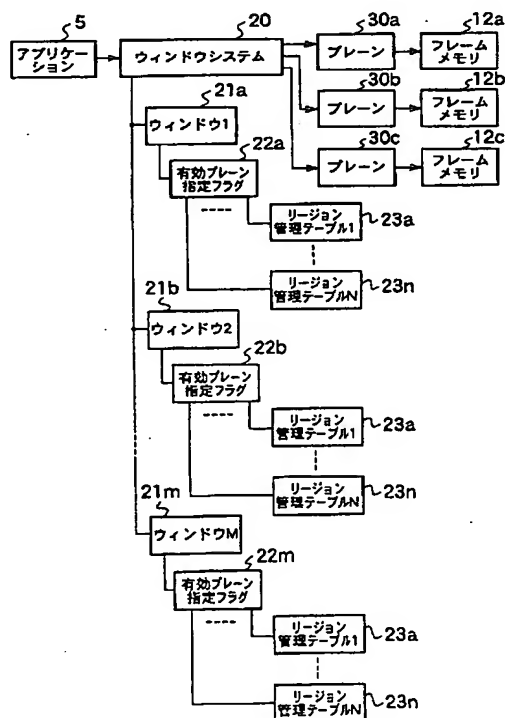
5C082 BA12 BB26 CA63 DA87 MM05

(54) 【発明の名称】 ウィンドウシステム

## (57) 【要約】

【課題】 複数のグラフィック・プレーンを容易に管理することができるウィンドウシステムを提供すること。

【解決手段】 アプリケーションプログラム (5) により生成される複数のウィンドウ (21a-21m) を描画する複数のプレーン (30a-30c) と、前記各プレーン (30a-30c) に対応し、前記各プレーンからのデータを記憶するフレームメモリ (12a-12c) とを有し、前記各フレームメモリ (12a-12c) からの出力データを配合或いは選択して表示装置 (16) に表示するウィンドウシステムであって、前記生成される複数のウィンドウ (21a-21m) 毎に、前記複数のプレーン (30a-30c) から描画を行うプレーン (30a-30c) を選択する有効プレーン指定手段 (22) を有することによる。



## 【特許請求の範囲】

【請求項 1】 アプリケーションプログラムにより生成される複数のウインドウを描画する複数のプレーンと、前記各プレーンに対応し、前記各プレーンからのデータを記憶するフレームメモリとを有し、前記各フレームメモリからの出力データを配合或いは選択して表示装置に表示するウインドウシステムであって、

前記生成される複数のウインドウ毎に、前記複数のプレーンから描画を行うプレーンを選択する有効プレーン指定手段を有することを特徴とするウインドウシステム。

【請求項 2】 請求項 1 に記載のウインドウシステムにおいて、

前記生成される複数のウインドウのうち描画が行われるプレーンを有効プレーンとして前記ウインドウが前記表示装置に表示される表示領域及び前記表示装置には表示されない非表示領域を矩形領域に分割し、各矩形領域の位置及びサイズを前記複数のウインドウの有効プレーン毎に管理するリージョン管理手段を有することを特徴とするウインドウシステム。

【請求項 3】 請求項 2 に記載のウインドウシステムにおいて、

前記ウインドウの各矩形領域の位置或いはサイズに変化が生じた場合、再描画が必要なプレーンと再描画する矩形領域の位置及びサイズを前記アプリケーションプログラムに通知することを特徴とするウインドウシステム。

## 【発明の詳細な説明】

## 【0001】

【発明の属する技術分野】 本発明は表示ウインドウを描画するための複数のプレーンを有し、各プレーンの表示データを配合或いは選択して表示装置に表示するウインドウシステムに関する。

## 【0002】

【従来の技術】 グラフィック・プレーンに複数のアプリケーション画面を出力する方法として、ウインドウシステムが挙げられる。このウインドウシステムは、アプリケーションプログラムに対して、ウインドウという仮想画面を提供し、その仮想画面を 1 枚のグラフィック・プレーンに重ね合わせるなどして同時に複数表示することができる機能を提供するソフトウェアである。ユーザは、マウス・ポインタ等でウインドウの枠や内部をクリックすると、それがアクティブなウインドウとなり、そのアプリケーションに対して、キーボードやマウスでの入力が可能になる。

【0003】 アプリケーションプログラムは、提供されたウインドウに対して描画処理を行えば、他のアプリケーションの画面描画処理等を意識する必要はない。

【0004】 このようなウインドウシステムは多くの商品が市場に出ているが、一例を挙げると、UNIX（登録商標）上で動作する X Window などがある。

【0005】 これらのウインドウシステムでは、アプリ

ケーションプログラムとの連絡のために、親ウインドウの ID と表示位置などの属性を引数として受け渡すウインドウ生成関数（例えば、X Window では、関数 `XccreateWindow()` である）を備えている。ウインドウシステムは、アプリケーションが実行するウインドウ生成関数に反応し、生成するウインドウの属性情報を格納するためのメモリを確保し、当該ウインドウの識別子をアプリケーションに返す、などの動作を行う。

【0006】 上記ウインドウ属性情報には、ウインドウの表示状態の情報も含まれ、ウインドウの表示領域のうち、実際に画面に表示されている領域と、他のウインドウが重なっているため非表示になっている領域の情報とが格納される。これらのウインドウ属性情報は、ウインドウの重なり合いやサイズなどを変更する際に、アプリケーションプログラムに対して、ウインドウの再描画を行うように再描画イベントを通知する際に用いられる。

## 【0007】

【発明が解決しようとする課題】 ところで、近年、複数のグラフィック・プレーンを持ち、これら複数のグラフィック・プレーンの内容をブレンド（配合）あるいは選択して表示するマルチ画面に対応した表示装置が開発されている。特に、テレビ受信機分野ではデジタル放送が開始され、これを受けて、今後、このようなマルチ画面の機能を備えたテレビ受信機が、家庭内デジタルネットワークにおける表示装置としての役割を期待されている。

【0008】 このようなマルチ画面の機能を備えた表示装置では、映像はもちろんのこと、静止画、コンピュータグラフィックス、テキストデータなど、DTV (Digital Television)、DVD (Digital Video Disc)、PC (Personal Computer) 等から供給される様々なデジタルコンテンツを、多画面で同時に表示させることが望まれている。

【0009】 しかしながら、従来のウインドウシステムでは、複数のグラフィック・プレーンについては考慮されていない。

【0010】 従って、例えば、プレーン A とプレーン B の 2 種類のグラフィック・プレーンを備えた表示装置においても、従来のウインドウシステムではどちらか一方のグラフィック・プレーンしか管理することができない。

【0011】 ウインドウシステムを 2 つ起動すれば、プレーン A とプレーン B の両方のグラフィック・プレーンを管理することができるものの、アプリケーションは起動される 2 つのウインドウシステムに対する処理をそれぞれプログラミングしなければならず、開発コストの増大を招く原因となる。

【0012】 本発明はこのような事情に鑑みてなされたものであり、複数のグラフィック・プレーンを容易に管理することができるウインドウシステムを提供することを目的とする。

## 【0013】

【課題を解決するための手段】上記課題を解決するために、本発明に係るウインドウシステムは、アプリケーションプログラムにより生成される複数のウインドウを描画する複数のプレーンと、前記各プレーンに対応し、前記各プレーンからのデータを記憶するフレームメモリとを有し、前記各フレームメモリからの出力データを配合或いは選択して表示装置に表示するウインドウシステムであって、前記生成される複数のウインドウ毎に、前記複数のプレーンから描画を行うプレーンを選択する有効プレーン指定手段を有することを特徴とする。

【0014】また、前記生成される複数のウインドウのうち描画が行われるプレーンを有効プレーンとして前記ウインドウが前記表示装置に表示される表示領域及び前記表示装置には表示されない非表示領域を矩形領域に分割し、各矩形領域の位置及びサイズを前記複数のウインドウの有効プレーン毎に管理するリージョン管理手段を有することを特徴とする。

【0015】また、前記ウインドウの各矩形領域の位置或いはサイズに変化が生じた場合、再描画が必要なプレーンと再描画する矩形領域の位置及びサイズを前記アプリケーションプログラムに通知することを特徴とする。

【0016】本発明の特徴によれば、アプリケーションプログラムは有効プレーンを選択的に指定することができるため、アプリケーション側およびウインドウシステム側の双方は指定された有効プレーンのみを管理すれば良い。

【0017】また、表示中の他のウインドウが移動したり閉じられたりして、今まで非表示であった領域が表示領域に変化する際にも、変化が生じたウインドウの有効プレーンがウインドウシステムから通知されるため、アプリケーションプログラムは通知されるプレーンに対してのみ再描画処理を施せば良い。

【0018】

【発明の実施の形態】以下、図面を参照しながら、本発明の実施の形態を詳細に説明する。

【0019】図1は本実施形態におけるウインドウシステムを動作させるためのハードウェアの構成例を模式的に示した図であり、図2はソフトウェアの構成例を模式的に示した図である。尚、図示した例では、3枚のグラフィック・プレーン30a~30cを備えたウインドウシステム20を示しており、グラフィック・プレーン30a~30c毎にそれぞれメモリコントローラ13a, 13b, 13cとフレームメモリ12a, 12b, 12cとを備えている。

【0020】CPU10にて実行されるウインドウシステム20は、バス11を介して、メモリコントローラ13a, 13b, 13cにより、各グラフィック・プレーン30a~30cのデータをフレームメモリ12a, 12b, 12cに書き込む。

【0021】グラフィック・プレーン30aのデータはメ

メモリコントローラ13aによってフレームメモリ12aから順次読み出されて、プレーン合成コントローラ14に転送される。同様にして、グラフィック・プレーン30b, 30cのデータはメモリコントローラ13b, 13cによってフレームメモリ12b, 12cから順次読み出されて、プレーン合成コントローラ14に転送される。

【0022】プレーン合成コントローラ14は各フレームメモリ12a, 12b, 12cから読み出されたデータを、ブレンド（配合）あるいは選択して、D/Aコンバータ15に転送する。D/Aコンバータ15は、プレーン合成コントローラ14から転送されてきたデータをアナログ信号に変換し、モニタ16に送る。

【0023】図2において、アプリケーションプログラム5は、ウインドウ生成関数を実行し、ウインドウシステム20に対してウインドウ21a~21mの生成を要求する。アプリケーションプログラム5は、ウインドウ21a~21mの生成を要求する際に、有効プレーン指定フラグ22a~22mにて、グラフィック・プレーン30a~30cの中から該ウインドウの描画を行うプレーンを有効プレーンとして選択的に指定する。また、アプリケーションプログラム5は、生成済みのウインドウ21a~21mに関して、ウインドウシステム20に対して描画、マップ等の要求も行う。

【0024】アプリケーションプログラム5からウインドウの生成、描画、マップ等の要求を受けたウインドウシステム20は、該ウインドウの有効プレーン毎に、該ウインドウの表示領域及び非表示領域をリージョン（REGION）と呼ばれる矩形領域に分割し、各リージョンの位置及びサイズを該ウインドウの有効プレーン毎に管理するリージョン管理テーブル23a~23nを作成する。そして、リージョン管理テーブル23a~23nの情報に基づいて、フレームメモリ12a, 12b, 12cに各グラフィック・プレーン30a~30cの表示データを書き込む。

【0025】図3（a）は本実施形態のウインドウシステム20におけるウインドウ生成関数CreateWindow()の指定例を示している。ウインドウ生成関数CreateWindow()は、引数として親ウインドウID（parent）と有効プレーン指定フラグ（plane）を備えており、アプリケーションプログラム5からウインドウシステム20に対して、親ウインドウID（parent）で指定されたウインドウを親として、新規にウインドウを生成するように要求する関数である。このウインドウ生成関数CreateWindow()は、出力として、新規に生成したウインドウのIDをアプリケーションプログラム5に返す。

【0026】有効プレーン指定フラグ（plane）は、図3（b）に例示するように、Nビットのビット列から構成されている。ビット0がプレーン0、ビット1がプレーン1、ビット2がプレーン2、・・・ビットNがプ

レーンNに対応しており、それぞれのビット値が「0」の場合は「無効」、「1」の場合は「有効」であることを示す。アプリケーションプログラム5は、ウインドウ生成時に複数あるグラフィック・プレーンのうち、生成するウインドウの描画を行うプレーンについて「有効」のフラグを立てる。

【0027】図4は、ウインドウの表示されている領域と表示されていない領域とを管理するためのメモリテーブルであるリージョン管理テーブル23の構成例を示している。このリージョン管理テーブル23は、ウインドウの表示領域を管理する表示リージョンテーブル24と、ウインドウの非表示領域を管理する非表示リージョンテーブル25とからなり、生成されるウインドウの有効プレーン毎に作成される。

【0028】表示リージョンテーブル24は、ウインドウの表示領域を表示リージョンと呼ばれる矩形領域に分割し、分割した表示リージョン数と、分割した表示リージョン毎に始点座標（X、Y）、サイズ（横幅、高さ）を記憶する。同様に、非表示リージョンテーブル25は、ウインドウの非表示領域を非表示リージョンに分割し、分割した非表示リージョン数と、分割した非表示リージョン毎に始点座標（X、Y）、サイズ（横幅、高さ）を記憶する。

【0029】図5は、グラフィック・プレーン30に2つのウインドウ40、50が表示されている例を示しており、ウインドウ50はウインドウ40の一部に重なっている。このウインドウ40の表示領域は、図6に例示するように、2つの表示リージョン41、42に分割することができ、ウインドウ40の非表示領域は、図7に例示するように、1つの非表示リージョン43とすることができ、図8に示すリージョン管理テーブル23として記憶される。

【0030】このように、ウインドウシステム20はウインドウの表示領域と非表示領域とをリージョン管理テーブル23にて常に管理することで、ウインドウの描画を行う際には、有効なプレーンの表示領域にのみ描画処理を行えば良い。尚、このリージョン管理テーブル23の各値は、ウインドウの位置やサイズ、他のウインドウとの重なり具合が変化するたびに更新される。

【0031】図9は、2枚のグラフィック・プレーン30、31を備えた表示装置で、グラフィック・プレーン30、31を有効プレーンとしたウインドウ40、グラフィック・プレーン30のみを有効プレーンとしたウインドウ50、グラフィック・プレーン31を有効プレーンとしたウインドウ60が表示されている様子を例示している。また、図10はグラフィック・プレーン30における上記各ウインドウの表示例、図11はグラフィック・プレーン31における上記各ウインドウの表示例を示している。

【0032】ここで、ウインドウ50がアンマップされ

た場合の、ウインドウシステム20の動作例を説明する。アンマップされるウインドウ50の有効プレーンは図10に示したグラフィック・プレーン30であるので、図12に示すように、同じくグラフィック・プレーン30を有効プレーンとするウインドウ40の重なり部分（非表示領域45）が表示領域に変化する。

【0033】図14は、ウインドウ50がアンマップされた場合の動作シーケンスを例示する図である。

【0034】まず、ウインドウ50に対して「閉じる」操作がなされると、ウインドウ50からウインドウシステム20に対してアンマップ要求が通知される（Step01）。

【0035】アンマップ要求を受けたウインドウシステム20は、所定のアンマップに係る処理を行い、ウインドウ50に対してアンマップを行った旨を通知する（Step02）。

【0036】次に、ウインドウシステム20は、ウインドウ40に対して再描画イベントを通知する（Step03）。

【0037】その際、再描画イベントには、図13に示すような再描画イベント情報26が付加される。この再描画イベント情報26は、再描画を要するグラフィック・プレーンと、再描画を要するリージョン情報の始点座標（X、Y）、サイズ（横幅、高さ）が含まれる。

【0038】再描画イベントを受けたウインドウ40は、再描画イベント情報26のデータに基づいて、グラフィック・プレーン30の指定された始点座標（X、Y）とサイズ（横幅、高さ）に該当する矩形部分について再描画を行った後、ウインドウシステム20に対して再描画完了を通知する（Step04）。

【0039】尚、ウインドウ40はグラフィック・プレーン31も有効であるが、アンマップされたウインドウ50はグラフィック・プレーン31が有効でないので、ウインドウ40のグラフィック・プレーン31に関する再描画イベントは通知されない。

【0040】また、ウインドウ60の有効プレーンはグラフィック・プレーン30ではないため、ウインドウ50がアンマップされても何の影響も受けない。

【0041】このように、アプリケーションプログラム5は有効プレーンを選択的に指定することができるため、アプリケーション5側およびウインドウシステム20側の双方は指定された有効プレーンのみを管理すれば良い。

【0042】また、表示中の他のウインドウが移動したり閉じられたりして、今まで非表示であった領域が表示領域に変化する際にも、変化が生じたウインドウの有効プレーンがウインドウシステム20から通知されるため、アプリケーションプログラム5は通知されるプレーンに対してのみ再描画処理等を施せば良い。

【0043】以上、本発明の実施形態について詳細に説

明したが、本発明は本実施例に限定されず、本発明の主旨を逸脱しない範囲において、種々の改良や変更を成し得るであろう。

【0044】例えば、本実施形態ではUNIXのX Windowをモデルに説明したが、本発明はこれに限定されず、他の形式のウィンドウシステムにも適用することができる。

【0045】

【発明の効果】本発明によれば、アプリケーションプログラムは有効プレーンを選択的に指定することができるため、アプリケーション側およびウィンドウシステム側の双方は指定された有効プレーンのみを管理すれば良い。

【0046】また、表示中の他のウィンドウが移動したり閉じられたりして、今まで非表示であった領域が表示領域に変化する際にも、変化が生じたウィンドウの有効プレーンがウィンドウシステムから通知されるため、アプリケーションプログラムは通知されるプレーンに対してのみ再描画処理等を施せば良い。

【0047】以上のことから、本発明によれば、複数のグラフィック・プレーンを容易に管理することができるウィンドウシステムを提供することができる。

【図面の簡単な説明】

【図1】本実施形態におけるウィンドウシステムのハードウェアの構成例を模式的に示した図である。

【図2】本実施形態におけるウィンドウシステムのソフトウェアの構成例を模式的に示した図である。

【図3】(a)は本実施形態におけるウィンドウシステムのウィンドウ生成関数CreateWindow()、(b)は有効プレーン指定フラグ(plane)の指定例を示したイメージ図である。

【図4】本実施形態におけるウィンドウシステムのリージョン管理テーブルの構成例を示した図である。

【図5】本実施形態において、グラフィック・プレーンに2つのウィンドウが表示されている例を示すイメージ図である。

【図6】図5に示したウィンドウの表示領域を2つの表示リージョンに分割した例を示すイメージ図である。

【図7】図5に示したウィンドウの非表示領域を1つの

非表示リージョンとした例を示すイメージ図である。

【図8】図6～図7に示した各リージョンの情報を格納したリージョン管理テーブルの具体例である。

【図9】本実施形態において、2枚のグラフィック・プレーンを備えた表示装置で、各ウィンドウが表示されている様子を例示した図である。

【図10】図9に示したグラフィック・プレーンの表示例を示した図である。

【図11】図9に示したグラフィック・プレーンの表示例を示した図である。

【図12】図10に示したグラフィック・プレーンにおいて、ウィンドウの重なり部分(非表示領域)が表示領域に変化した例を示した図である。

【図13】本実施形態において、再描画イベントの通知時に付加される再描画イベント情報の具体例を示す図である。

【図14】本実施形態における再描画イベント時の処理シーケンスを例示する図である。

【符号の説明】

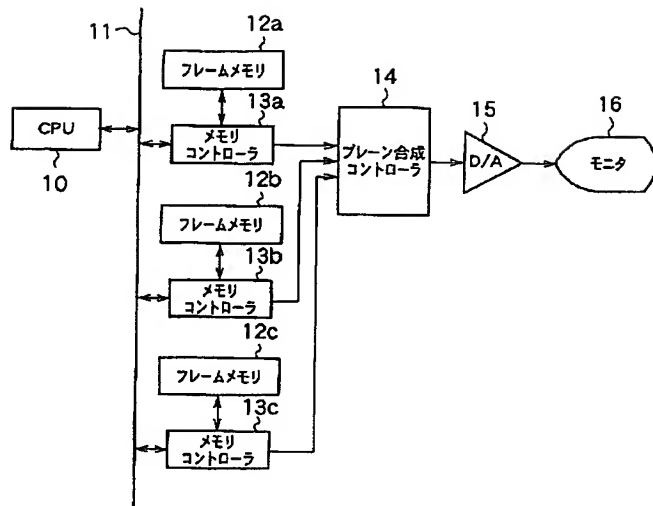
5...アプリケーションプログラム  
10...CPU  
11...バス  
12a～12c...フレームメモリ  
13a～13c...メモリコントローラ  
14...プレーン合成コントローラ  
15...D/A(デジタル/アナログ変換器)  
16...モニタ  
20...ウィンドウシステム  
21a～21m...アドレス保持部  
22a～22m...有効プレーン指定フラグ  
23, 23a～23m...リージョン管理テーブル  
24...表示リージョンテーブル  
25...非表示リージョンテーブル  
26...再描画イベント情報  
30, 30a～30c, 31...グラフィック・プレーン  
40, 50, 60...ウィンドウ  
41, 42...表示リージョン  
43, 45...非表示リージョン

【図13】

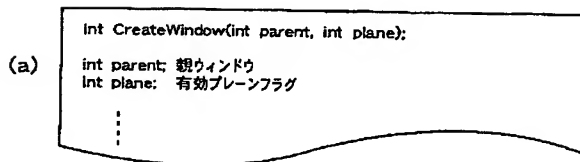
(再描画イベント情報26)

再描画プレーン:30
X座標:200
Y座標:300
幅:700
高さ:300

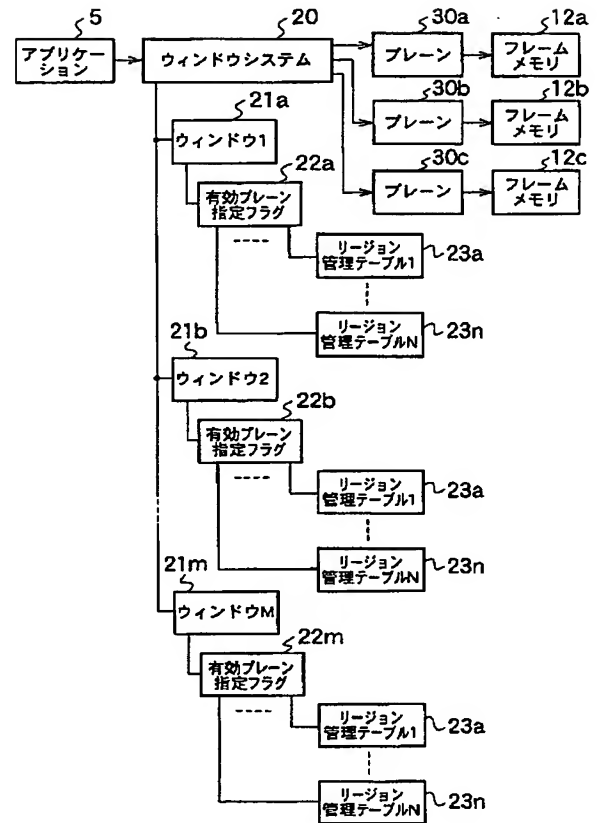
【図1】



【図3】



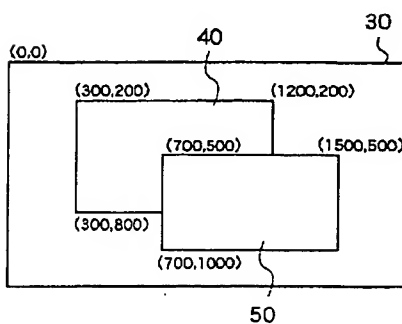
【図2】



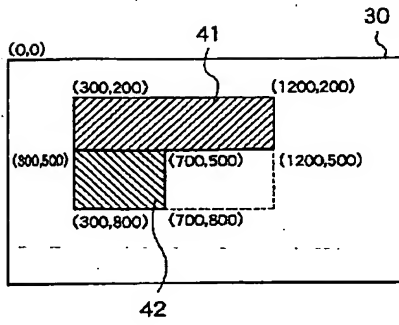
【図4】



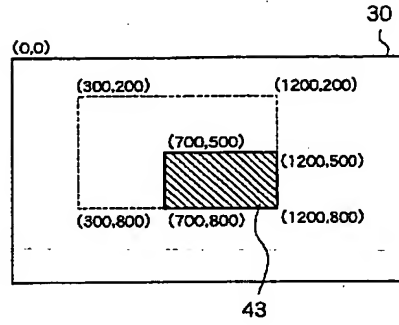
【図5】



【図 6】



【図 7】

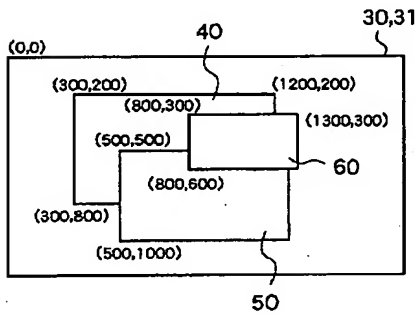


【図 8】

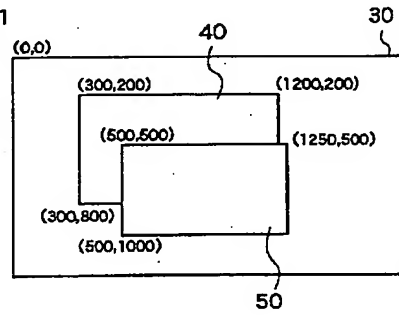
リージョン管理テーブル23

表示リージョンテーブル24		非表示リージョンテーブル25	
表示リージョン数 (R1+1):2		非表示リージョン数 (R2+1):1	
表示リージョン0	X座標:0	非表示リージョン0	X座標:400
	Y座標:0		Y座標:300
	横幅:900		横幅:500
	高さ:300		高さ:300
表示リージョン1	X座標:0		
	Y座標:300		
	横幅:400		
	高さ:300		

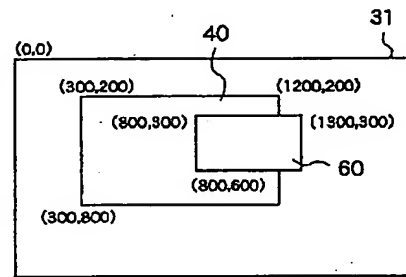
【図 9】



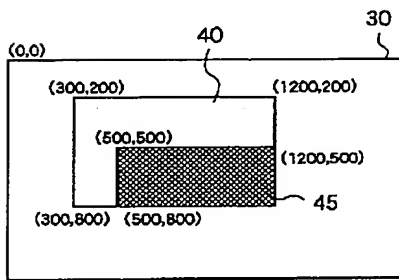
【図 10】



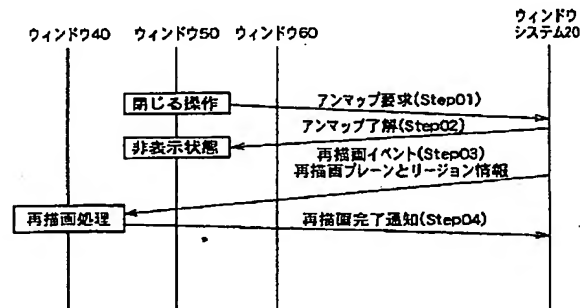
【図 11】



【図 12】



【図 14】



**THIS PAGE BLANK (USPTO)**